DSA Assignment

ch. Rishitha
AP19110010441
CSE-F

1)

```
#include<stdio.h>
# define NUM30
void bubblesort(int array[], int size)

{
 for (int i =0; j <size-1; ++i)
 for (int j =0; j<size-i-1; ++j)
 {
   if (array[j] <array[j+1])
   {
     int temp =array[j];
     array[j] = array[j+1];
     array[j+1] =temp;
   }
 }
}
void display(int array[], int size]
{
  for(int i=0; i<size; ++i)
  { printf("%d", array[i]);
  }
  printf("\n");
}
int binarysearch(int array[], int l, int r, int x){
if(r>=l){
    int mid=l+(r-l)/2;
    if (array[mid]==x){
      return mid;
}
elseif(array[mid] > x){
return binarysearch (array,l,mid-1,x);
}
 else{
   return binarysearch(array,mid+1,r,x);
   }
 }
 return -1;
}
void sumandproduct(int array[]){
 int loc1,loc2;
  printf("Enter location 1:");
   scanf("%d",&loc1);
  printf("Enter location 2:");
   scanf("%d",&loc2);
   printf("Sum of elements in positions %d and
        %d is:%d\n",loc1, loc2,array[loc1-1]+
                 array[loc2-1]);
 Printf ("Products of elements in positions %d and
        %d is:%d\n",loc1,loc2,array[loc1-1]*
          array[loc2-1]);
 }
int main()
{
 int a[NUM],size,k,r,result;
 printf("Enter no of elements of array:");
 scanf("%d",&size);
 for(k=0;k<size;k++)
 {
```

```
printf("Enter the %dth element :",k+1);
Scanf ("%d", &a[k]);  ]
}
printf (Given array :\n");
display (a,size);
bubble sort (a,size);
printf("Sorted Array in Descending Order :\n");
display(a,size);
printf("a)\n");
printf("Enter the elements to search:");
Scanf ("%d", &r);
result = binary search (a,0,size-1,r);
if(result==-1){
 Printf (" %d element is not found in sorted
                 array\n",r);
}
else{
   printf ("%d element is found in sorted array
          at location %d\n",r,result+1);
 }
printf("b)\n");
sum and product(a);
return0;
}
```

②
```
#include<stdio.h> using name space std;
#define n4
void merge arrays (int arr1[ ], int arr2[ ], int n2,
                                   int arr3[])
{
int i=0,j=0, k=0;
while (i<n1&&j<n2)
{
```

```
if(arr1[i]<arr2[j])
   arr3[k++]=arr1[i++];
else
   arr3[k++] = arr2[j++];
}
while
   (i<n1)
    arr3[k++]= arr1[i++];
 while (j<n2)
    arr3[k++]= arr2[j++];
 void print array(int arr[ ], int size),
 {
  for(int i=0;i<size;i++)
  Count << arr[i]<<" ";
                                              int j, int
 void merge k arrays(int arr[ ][n],int i, output[ ])

 {
   if (i==j)
   {
    for (int i=0;p<n;p++)
     [p]= arr[1][p];
     return;
   }
   if (j-1==1)
   {
    merge Arrays(arr[i],arr[j],n,n);
    return;
   }
```

Scanned with CamScanner

3) Insertion Sort : Insertion sort works by inserting the set values in the existing sorted file. It constru-cted the sorted array by Inserting a single Element at a time. This process continouous until whole array is sorted in some order. The primary concept behind Insertion sort is to Insert each item into its appropriate place in the final list. The Insertion sort method save an Effective amount of memory.

Advantages of Insertion Sort:

* Easily implemented and very efficient when used with small sets of data.

* The additional memory space Recuirement of Inserting sort is less (i.e O(1))

* It is considered to be Live sorting techniques as the list can be sorted as the new Elements are received.

* It is faster than other sorting algorithms.

Example :

25	15	30	9	99	20	26
15	25	30	9	99	20	26
15	25	30	9	99	20	26
9	15	25	30	99	20	26
9	15	25	30	99	20	26
9	15	20	25	30	99	26
9	15	20	25	26	30	99

Selection Sort :

Definition :

The selection Sort perform sorting by searching for the minimum value number and placing it into the first of Last Position according to the order. The process of searching the minimum key and placing it in the proper position is continued until the all the Elements are placed at right position.

Advantages of selection sort:

* suppose an array ARR with N elements in the memory.
* Simple to understand the sorting of Elements doesn't depend on the intial arrangement of the Elements

Example:

	0	1	2	3	4
1 →	17	16	3	15	6
	17	16	3 ↑	15	6
	min		LOC		
2 →	3	16	17	15	6
		min			LOC
3 →	3	6	17	15	16
			min	LOC	
4 →	3	6	15	17	16
				min	LOC
5 →	3	6	15	16	17

4)
```
# include <stdio.h>
# define NUM 30
void bubblessort (intarray[], int size)
{
  for (int i=0; i<size-1; ++i)
  for (int j=0; j<size-i-1;++j)
   {
    if (array [j] > array[j+1])
    {
     int temp = array [j];
     array [j] = array [j+1];
     array [j+1] = temp;
    }
   }
}

void display (int array[], int size)
{ for (int i = 0; i<size; ++i){
   printf ("%d", array[i]);
 }
  printf ("\n");
 }
 void smandproduct (int array[], int size)
 {  int sum=0, product = 1;
    for (int i=0; i<size; i=i+2) {
      sum = sum+ array[i];
```

```
for(int j=1; j<size ;j=j+2){
  product= product * array[j] ;
}
 printf (" sum of elements in odd position: %d\n",sum);
 printf (" product of elements in even position: %d\n",product);
}
 void divisible( int array[], int size)
{
  int m;
   printf (" Enter value of m: ");
   scanf (" %d",&m);
   printf (" Elements of array divisible by %d are :\n",m);
   for(int i=0; i<size;i++){
      if (array [i] % m ==0){
      printf (" %d", array[i]);
    }
} }
 int main( )
 {
   int a[NUM], size, k ;
   printf ("Enter no of elements of array : ");
   scanf (" %d", &size);
   for (k=0; k< size; k++)
   {
     printf (" Enter the %dth element: ", k+1);
   } scanf (" %d" , &a[k]);
```

```
printf ("Given array: \n");
 display (a, size);
 bubblesort(a, size);
 printf (" sorted Array in Ascending order : \n");
 display (a, size);
 printf ("a\n");
 printf (" sorted Array in Alternate order : \n");
 alternate (a, size);
 printf ("b\n");
 sum and product (a, size);
 printf ("c\n");
 divisible (a, size);
 return 0;
}.
```

(5) Recursive Program for binary search.

```
# include<stdio.h>
void binary_search(int [], int, int, int);
void bubble_sort (int [], int);
int main()
{
  int key , size, i;
  int list[25];
  printf (" Enter size of a list :");
  scanf ("%d", &size);
```

```
printf (" Enter elements");
for (i=0; i<size; i++)
{
 scanf ("%d", &list[i]);
 bubble_sort (list, size);
 printf ("\n")
 printf ("Enter key to search \n");
 scanf ("%d", &key);
binary_search (list, 0, size, key);
}
void bubble_sort (int list[], int size)
{
  int temp, i, j;
  for (i=0; i<size; i++)
  {
   for (j=i; j<size; j++)
   {
    if (list [i] > list(j])
   {
    temp = list [i];
    list[i] = list [j];
    list [j] = temp;
   }
  }
 }
}
```